AFRL-IF-RS-TR-2002-293
Final Technical Report
November 2002

MULTI-COMMUNITY CYBER DEFENSE (MCCD)

Boeing Phantom Works

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. H561

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2002-293 has been reviewed and is approved for publication.

APPROVED:

KEVIN DAMP, Capt., USAF
Project Engineer

FOR THE DIRECTOR:

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 2002	Final Oct 99 – Dec 01

4. TITLE AND SUBTITLE
MULTI-COMMUNITY CYBER DEFENSE (MCCD)

5. FUNDING NUMBERS
C - F30602-99-C-0181
PE - 62301E
PR - H561
TA - 10
WU - 01

6. AUTHOR(S)
Randall Smith

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Boeing Phantom Works
Engineering Technology
PO Box 3999
Seattle Washington 98124-2499

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFGB
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2002-293

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Capt. Kevin Damp/IFGB/(315) 330-7858/ Kevin.Damp@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

This program developed and demonstrated automated technologies enabling security devices to cooperatively respond to network intrusions across small-to-very-large-scale networks while spanning administrative domains. Theatre-wide network defense is achieved by enabling cooperative intrusion tracking and by sharing attack-related information and response recommendations between neighboring domains. This effort extended the Intruder Detection and Isolation Protocol (IDIP), which uses intrusion detection systems and cooperating boundary controllers within a single administrative domain to track network intruders to their origin and dynamically change network-level access control policies to stop the attacks in real-time. The focus of this effort was to develop, implement, and demonstrate enhancements to IDIP extending the intrusion tracing, response, and reporting mechanisms over organizational boundaries, enabling organizations to control the intrusion-related information they share and the degree of cooperation they provide, and to provide a policy-driven service that recommendations changes to local cooperation policies based on the threat status of neighboring communities. A real-time, strategic-level intrusion correlation engine was developed and demonstrated using the inter-community information sharing services to receive anomaly reports from neighboring communities, enabling early detection of widespread, stealthy scanning activities that would otherwise go undetected.

14. SUBJECT TERMS
Computer Network Defense, Intrusion Detection System

15. NUMBER OF PAGES
41

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

Abstract

This report documents the final results of Contract F30602-99-C-0181 titled "Multi Community Cyber Defense." This program developed and demonstrated automated technologies enabling security devices to cooperatively respond to network intrusions across small- to very-large-scale networks while spanning administrative domains. Theatre-wide network defense is achieved by enabling cooperative intrusion tracking and by sharing attack-related information and response recommendations between neighboring domains, subject to administratively established constraints.

This effort extended the Intruder Detection and Isolation Protocol (IDIP), which uses intrusion detection systems and cooperating boundary controllers (filtering routers or firewalls) within a single administrative domain to track network intruders to their origin and dynamically change network-level access control policies to stop the attacks in real-time. The focus of this effort was to develop, implement, and demonstrate enhancements to IDIP extending the intrusion tracing, response, and reporting mechanisms over organizational boundaries, enabling organizations to control the intrusion-related information they share and the degree of cooperation they provide, and to provide a policy-driven service that recommendations changes to local cooperation policies based on the threat status of neighboring communities. A real-time, strategic-level intrusion correlation engine was developed and demonstrated using the inter-community information sharing services to receive anomaly reports from neighboring communities, enabling early detection of widespread, stealthy scanning activities that would otherwise go undetected.

Prototype demonstrations and experimentation have shown the techniques described in this report to be effective at identifying and containing attacks that (1) hide the true adversary's identity by traversing multiple administrative boundaries, (2) avoid current detection schemes by using low-frequency probes against widely-distributed resources, and (3) neutralize traditional defenses by using a large number of distributed resources to execute attacks. This report describes the results of developing and demonstrating the MCCD-based components.

Multi Community Cyber Defense

1.0 INTRODUCTION

The objective of this effort was to investigate, develop, evaluate, integrate, document, and demonstrate technology for security devices that cooperatively respond automatically to network intrusions across small- to very-large-scale networks of networks spanning numerous administrative domains. The specific objectives were to develop, demonstrate, and assess:

a. Intrusion correlation techniques and tools that scale up to regional and national levels.
b. A trust model for intrusion detection and response (IDR) across disjoint administrative domains, with techniques for assessing trust.
c. Capabilities required for survivable, cooperating IDR systems across organizational boundaries.

The approach taken was to develop the required technology that included (1) an operational concept, (2) specific mechanisms, (3) a policy language suitable for defining organizational relationships, and (4) implementations of an edge boundary controller, enhanced management services, and an anomaly sensor and correlator. The operational concept was then validated through demonstrations of the resulting technology.

Enhancements to the intrusion detection and response infrastructure were designed to enable the following functions:

a. Track down the attack launch point across networks spanning multiple administrative domains (given that the intruder had successfully compromised a host outside of the detecting domain).
b. Maintain local autonomy by providing administrators with technical mechanisms to manage the information shared with, and services provided to neighboring domains during an intrusion investigation.
c. Dynamically monitor conditions effecting trust relationships with neighboring domains, and issue advisories when conditions change.
d. Monitor anomalous activities within a domain, and correlate those events that might indicate widespread, low frequency adversary intelligence gathering activities.
e. Provide mechanisms for sharing anomaly reports, intrusion alerts, and investigation reports with neighboring domains.

1.1 Background

Our growing dependence on information systems leaves the U.S. vulnerable to large-scale cyber attacks from other countries or terrorists intent on causing widespread disruption. A strategy of national cooperation between Government and the private companies that own critical national infrastructures is needed for defense against such attacks to recognize when multiple parts of the

infrastructure are simultaneously under attack, and to respond cooperatively. This requires an infrastructure that supports strategic intrusion correlation and automated response, tools to perform strategic intrusion correlation, and mechanisms that enable incident-related information sharing across corporate and Government boundaries without creating additional security risks by exposing internal capabilities or potential vulnerabilities. These mechanisms must protect sensitive information and allow organizations to establish and manage trust relationships with neighboring organizations.

Multi-staged attacks can be used to achieve strategic goals by identifying critical targets while avoiding detection by hiding the true source of the attack, distributing the attack agents, and by reducing the attack traffic below detection thresholds. Figure 1 illustrates a typical multi-staged network attack that traverses multiple networks and organizations. The attack is initiated and controlled from an attack host, located somewhere on the Internet. The true source of the attack is disguised by laundering the attack path through one or more compromised stooge hosts, often in different networks. Even if the victim (or an intrusion detection system located near the victim) detects the attack, it will be unable to trace the attack beyond the nearest stooge host without that organization's cooperation.

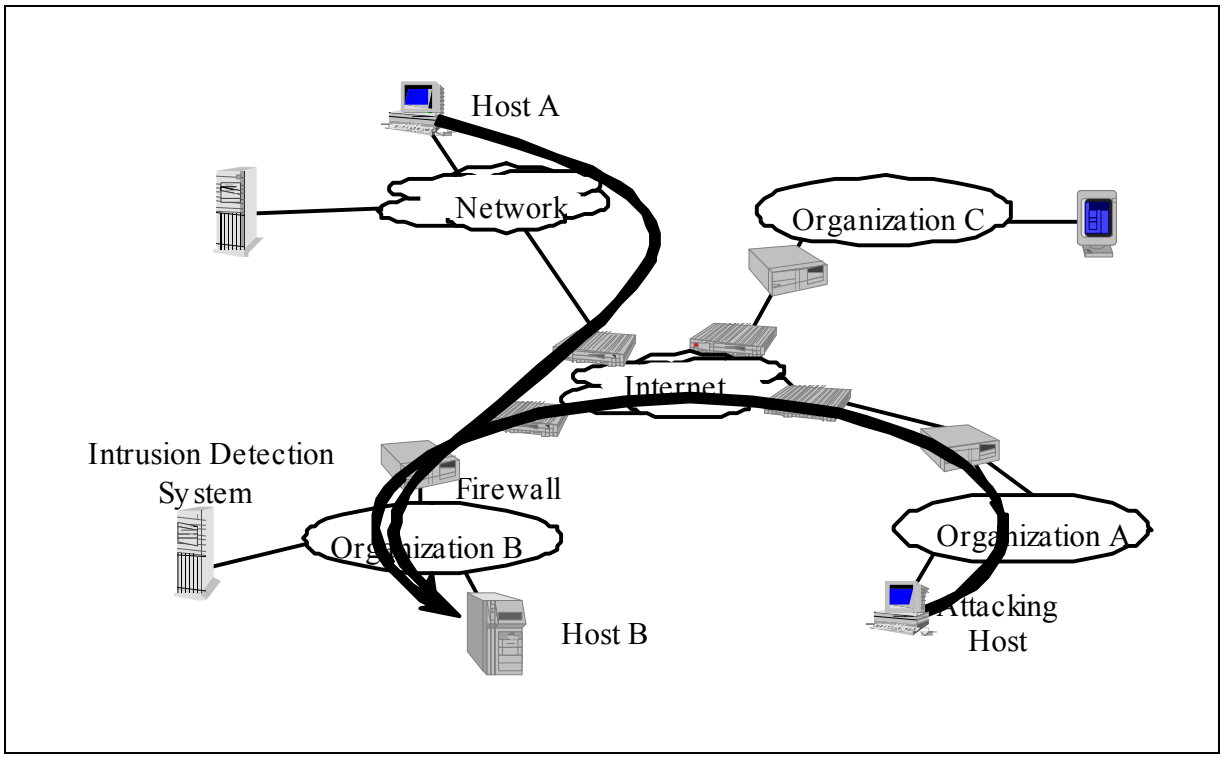

Figure 1 Multi-Staged Network Attacks

Existing defensive practices are ineffective at discovering and isolating multi-staged attacks. While the attack is occurring at computer speeds, the defense is a manual process requiring administrators to contact someone responsible for each compromised computer along the attack path until the true source is located. This process often fails due to a single unresponsive (or nonexistent) administrator in the path. Even when cooperative administrators can be reached, distributed attack techniques (e.g., distributed denial of service attacks) make it impractical to contact all attacking computers rapidly enough to prevent serious damage. Effective network defense against strategic attacks will require some degree of automated cooperation and collaboration between affected domains, with automated tracing mechanisms and coordinated responses.

The Intruder Detection and Isolation Protocol (IDIP) provides boundary controllers (filtering routers or firewalls) in the attack path with sufficient information to permit tracking intruders to the network component within an organization that is closest to the attack source, typically the corporate firewall, and can block the traffic nearest the source. IDIP has been demonstrated to be effective at stopping attacks within a single administrative domain, but does not trace attacks beyond the local domain boundary; intruders outside the detecting domain are still able to launch new attacks against resources in the original or neighboring domains.

Effective intrusion response to multi-staged network attacks that traverse organizations requires mechanisms that (1) enable organizations to securely share IDIP trace and report messages, and (2) establish local cooperation policies that allow each organization to withhold sensitive intrusion-related information and restrict automated intrusion trace and response actions. The Multi Community Cyber Defense (MCCD) services extend the IDIP concepts, enabling neighboring organizations, each running their own internal IDIP service, to cooperate when tracing attacks originating outside their organization. The MCCD reporting service allows sharing intrusion trace path reports and intrusion response actions. Reports of anomalous activities can be shared with neighboring organizations, or regional or national analysis centers, where they can be aggregated to provide early detection of reconnaissance activities, and a more complete view of wide-spread attacks.

1.2 Approach

MCCD extends the results of DARPA's Information Survivability program (Common Intrusion Detection Framework – CIDF [11] and Intruder Detection and Isolation Protocol – IDIP [5]) to enable cooperation between neighboring communities in intrusion tracing, response, and reporting, while maintaining administrative control over their own information and services. With this extended defensive framework, information about anomalous activities can be collected from multiple communities, and correlated to yield early detection of widely distributed, low-frequency reconnaissance activities.

Figure 2 shows the flow of information in a distributed, hierarchical intrusion detection and response system. Each community contains its own intrusion detection system (IDS) watching for signs of intrusions. Information from detectors is sent to management components (DC/Spice/TM) within the local community for further analysis and is also sent to neighboring components within its community in an attempt to locate the source of the activity and take local actions intended to stop the attack. Cooperation from neighboring communities is needed to respond to attacks originating outside the detecting community. In this case, the source of the activity is traced to the local community's edge boundary controller (EBC) where information may be sent to neighboring EBCs who can continue the trace. An administrator uses a local management console to define the degree of cooperation and sharing enforced at each local EBC.

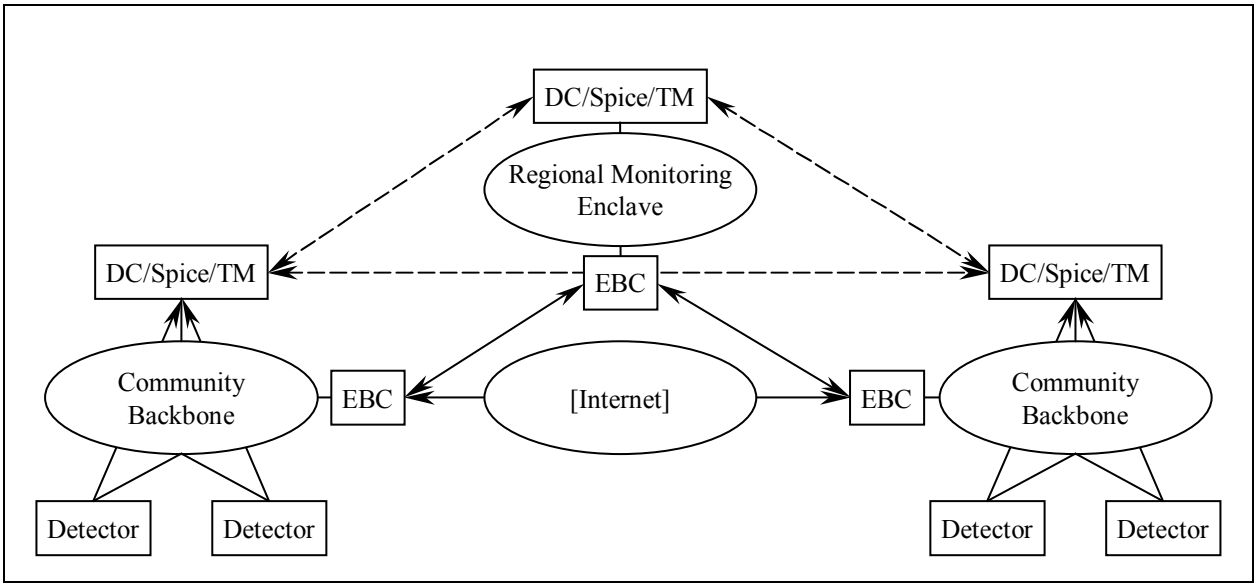

Figure 2 MCCD Information Flow

The MCCD management channel is used to send intrusion reports to the detecting community, and send correlation information to analysis centers where reports from multiple communities can be further analyzed. Situation displays may be updated based on intrusion reports, and correlation results produced at a regional and national level may cause intrusion alerts to be sent to subordinate systems warning of wide-scale attacks. These alerts may be used at the lower levels to generate automated response or updates to situation displays.

1.3 Summary of Accomplishments

The efficacy of the MCCD cooperative intrusion response concept, protocol, and mechanisms was validated via two technology demonstrations and an experiment. In the first demonstration verified that the enhanced communications infrastructure successfully traced and blocked attacks originating in neighboring communities (with or without administrator authorization, based on

local policy), and that the intrusion correlation engine could detect a slow, stealthy port scan. The second demonstration used a more complex architecture to illustrate a complete cooperative intrusion response infrastructure that involved tracing attacks through multiple communities with differing policies. An enhanced administrative service to aid with establishing mutually acceptable policies, a monitor to detect changes in attacker activity in neighboring communities, analysis of anomaly reports shared with neighboring communities, and integration with a prototype course of action generation tool were also demonstrated.

The MCCD prototype satisfied the program objectives in the following areas:

a. Intrusion correlation techniques and tools that scale up to regional and national levels. The strategy of performing analysis and correlation of anomalous activities at each hierarchical level (sensor, local correlator, regional or national analysis center) effectively reduced the workload of each component. Common network activity is analyzed and quickly discarded, while anomalous activities are maintained. Higher levels receive reports of identified intrusion activities (scans) and periodic anomaly state information requiring further correlation. This hierarchical analysis approach minimizes the network bandwidth and processing requirements at each level.

b. A trust model for intrusion detection and response (IDR) across disjoint administrative domains, with techniques for assessing trust. MCCD uses a generic trust event correlator (TEC) and a trust event specification language for describing trust change conditions. The trust event correlator monitors events from an input stream, trying to match patterns defined by the trust model. When a match is found, a new event is generated and placed back on the stream where it could be used as the input to another correlator. Custom input/output software provides the interface to the event stream, and is used to generate trust relationship change recommendations based on TEC events.

c. Capabilities required for survivable, cooperating IDR systems across organizational boundaries. MCCD addresses three key cross-organizational issues: (1) intrusion trace and response services, (2) message-layer encryption, and (3) local control of IDR information and services. MCCD extensions to IDIP provide intrusion trace, response, and report services between cooperating organizations, enabling attacks to be traced across organizational boundaries. IDIP uses a Neighborhood Key Information Distribution (NKID) service to insure authentication, privacy, and message integrity for IDR information exchanges within a local community. MCCD extends NKID to provide these services for MCCD information exchanged between communities. Administrators manage IDR services and information through policies that can be developed locally, or can be negotiated with neighboring communities to achieve mutually acceptable sets of services. MCCD components at community boundaries enforce the administrative policies on shared information and services.

1.4 Scope

The scope of this effort was to investigate issues related to multi-community cooperation in response to strategic cyber attacks, including correlation at the regional and national levels, determining trust among disjoint organizations, enabling attack-related information flow between organizations, and building a survivable detection and response infrastructure.

This final technical report summarizes the Multi Community Cyber Defense project results, including–

a. Summary of the MCCD architecture and the multi-community mechanisms implemented, including the operational concept, policy enforcement mechanisms, trust evaluation, and correlation techniques.

b. Summary of project accomplishments and capabilities developed.

c. Description of experimentation and proof-of-concept demonstrations conducted to validate techniques and demonstrate capabilities.

d. Summary of the project, including lessons learned and recommended future work to better exploit this technology in operational environments.

2.0 MCCD SUMMARY

Multi Community Cyber Defense (MCCD) addresses a number of policy issues that arise from extending the Intruder Discovery and Isolation Protocol (IDIP) to Intrusion Detection and Response (IDR) systems that span multiple communities, or administrative domains. The following sections provide a summary of MCCD architecture and the multi-community mechanisms implemented, including the operational concept, policy enforcement mechanisms, trust evaluation, and correlation techniques

There are several IDIP device types, including intrusion detection components, boundary controllers (i.e., firewalls and routers), network management (the discovery coordinator), and end systems. MCCD extends the IDIP network management functionality, defines a new class of boundary controller, introduces the concept of trust management, and provides a correlator to analyze information from one or more communities.

To help understand how MCCD operates, the following terms require definition:

a. **IDIP Neighborhood** – A collection of adjacent IDIP components (i.e., two IDIP components are neighbors if they do not have an IDIP component between them).
b. **Discovery Coordinator** – An IDIP component that receives attack descriptions and descriptions of each IDIP node's response, and potentially directs the overall system response. Each IDIP node reports to a single discovery coordinator.
c. **Community** – A set of IDIP neighborhoods sharing a common Discovery Coordinator. Each IDIP community may comprise multiple networks, with multiple boundary controllers and intrusion detection systems spread across these networks; however, they fall under the control of a single administrative authority.
d. **Edge Boundary Controller** – A Firewall or Guard (classification domain boundary controller) at or near a community's perimeter that communicates with EBCs in other communities. All MCCD messages exchanged between two communities is routed through their respective EBCs.
e. **MCCD Neighborhood** – A collection of adjacent, independent IDIP communities (i.e., two MCCD communities are neighbors if they do not have an MCCD community between them).

Figure 3 shows a typical relationship between two MCCD communities. Within each community, the intrusions are reported to the discovery coordinator, and community-level control is received from the discovery coordinator. Each discovery coordinator corresponds to an administrative domain, and discovery coordinators have reporting relationships that follow the relationships of the corresponding administrative domains. Between communities, the intrusions are traced by edge boundary controllers, each enforcing policies established by their respective discovery coordinator.

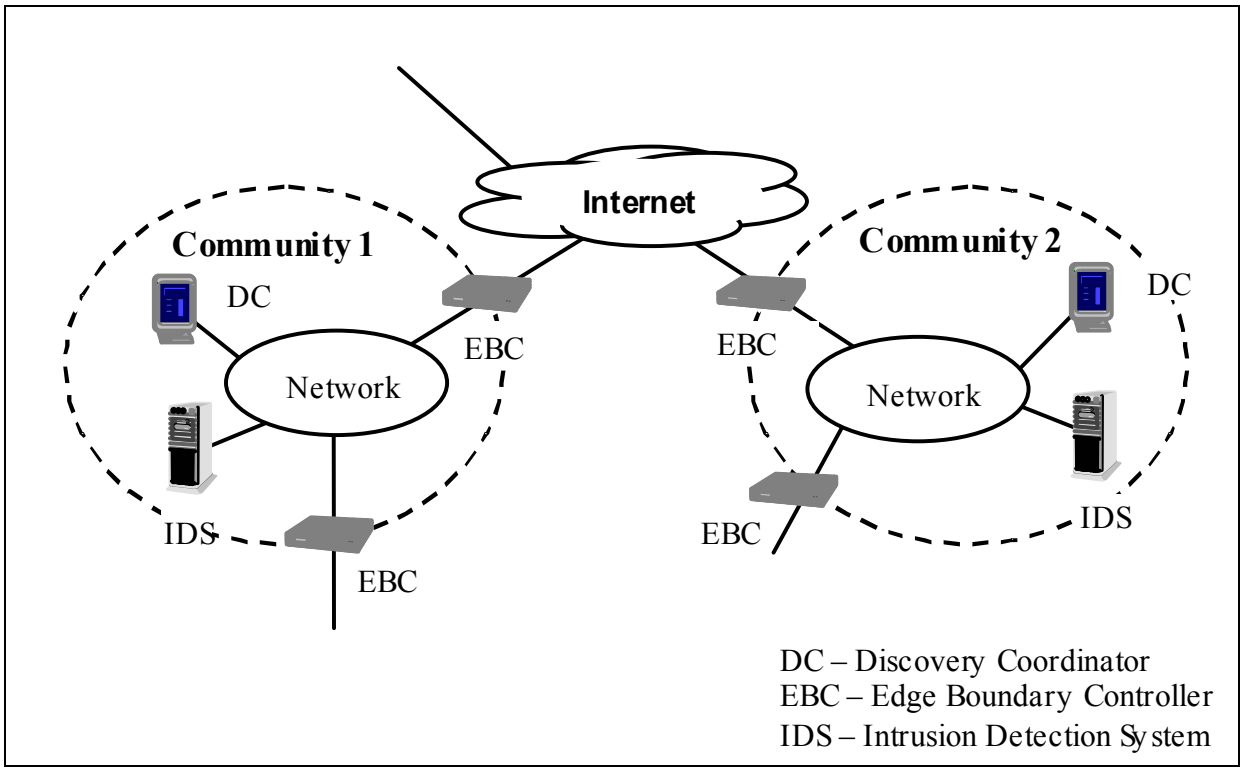

Figure 3 Typical MCCD Communities

The following sections summarize the MCCD operational concept and architectural components.

2.1 MCCD Operational concept

MCCD can detect stealthy, widespread intruder activities and track intrusions across network domain boundaries, temporarily blocking further activity from the intruder if that activity is interfering with the system's mission. This is shown in Figures 4 through 7. An attacker uses workstations he has compromised (by inserting Trojan horse software) to map networks of interest looking for vulnerable services that could be compromised. MCCD Sensors in those networks detect the unusual activities and report them to local MCCD correlators. These correlators share aggregated reports with regional (or national) centers where the widespread, stealthy scanning activities are identified.

The attacker uses one or more of the compromised workstations to launch an attack against a critical resource located in the target network (Figure 4), using spoofed addresses to hide the true location of the attack source. Even though the addresses are forged, MCCD components locate the attacker's network, temporarily blocking selected traffic from the attacker's packet stream, if needed, to protect local resources (Figure 5).

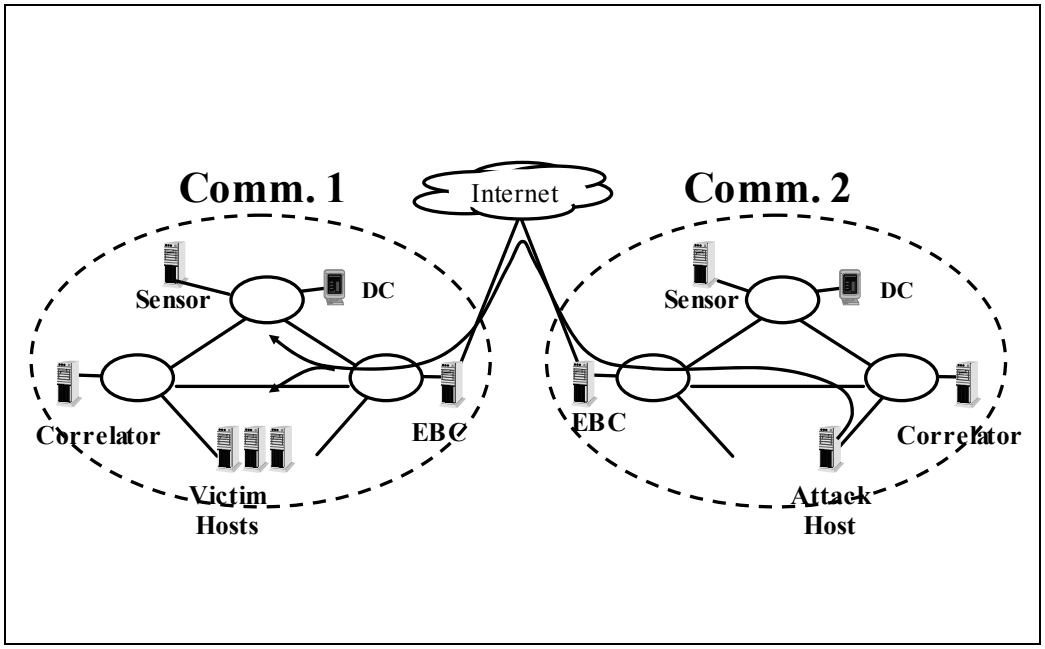

Figure 4 Intruder Scans and Attacks from Compromised Host

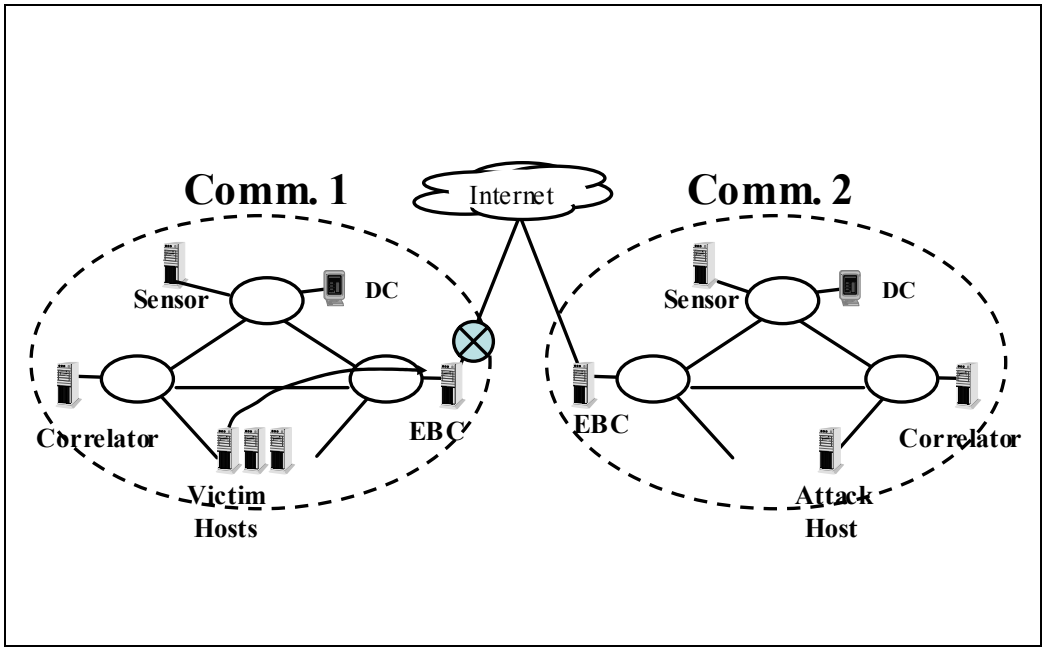

Figure 5 Detecting, Tracing, and Limited Blocking within a Community

If the attacker's network also implements MCCD, then the edge boundary controller in the detecting network asks the neighboring community for cooperation in tracing and blocking the attack (Figure 6). The degree of cooperation, the information shared, and the level of automation vs. administrative review is determined by individual organizational policies enforced at each edge boundary controller. The attacking community traces the attack path, locates the compromised workstation, and temporarily blocks selected traffic from the attacker's packet stream, if needed (Figure 7).

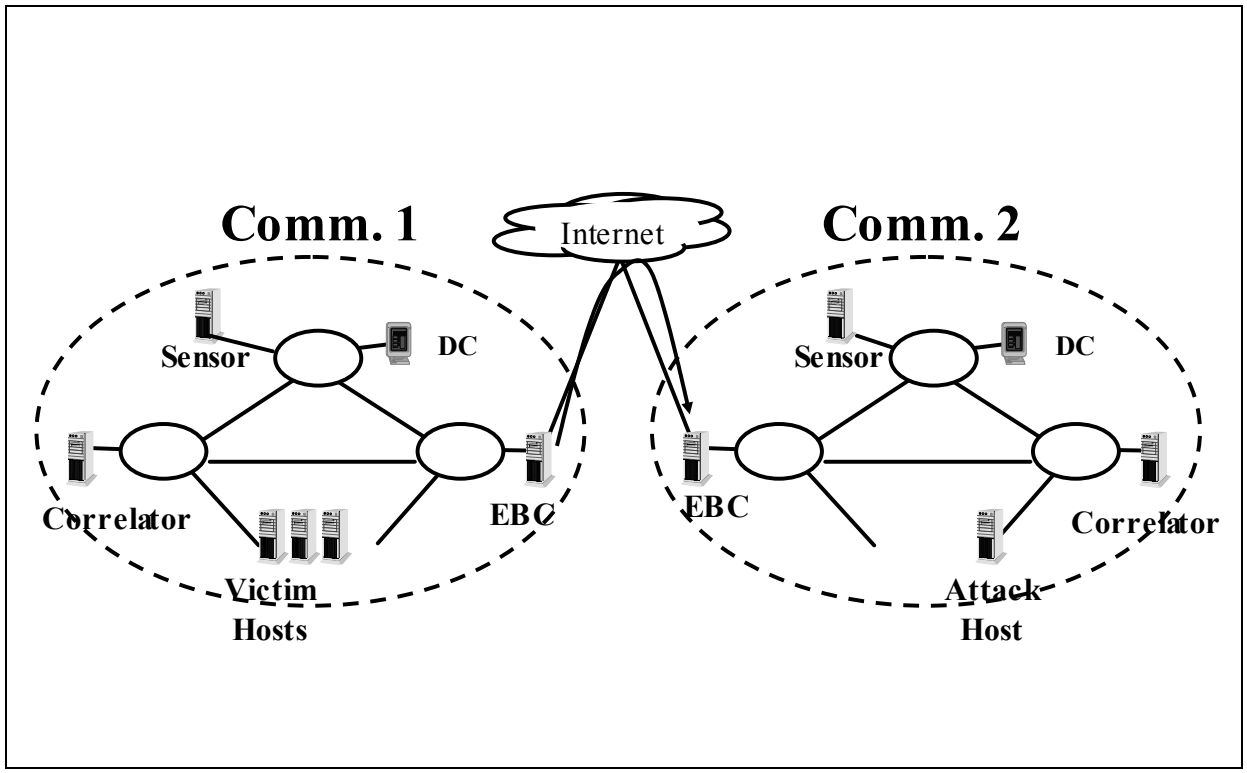

Figure 6 Trace Cooperation between Two communities

MCCD components located along the attack path report their findings to local management stations where administrators can determine and implement appropriate long-term actions (change network blocking rules, take the compromised workstation off-line until it is cleaned, etc.). Policies between neighboring communities are based on local perceptions of the trustworthiness of resources in remote communities. Trust monitors use aggregated reports to detect changes in the level of attacker activities in neighboring communities, and use this information to recommend changes to the degree of automated cooperation and sharing provided to each neighbor.

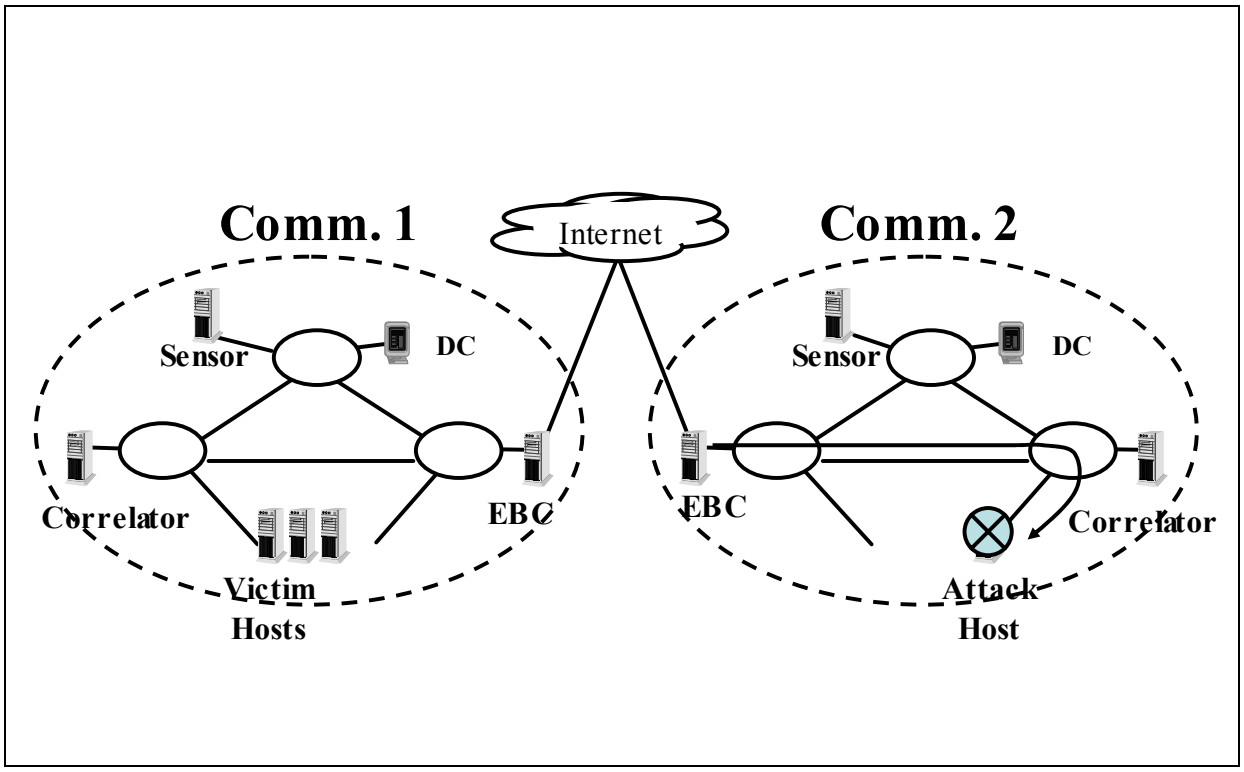

Figure 7 Remote Community Tracing and Blocking

2.2 IDIP Services and Applications

MCCD messages are transmitted over the IDIP backplane [4], a communications protocol that provides end-to-end encryption and authentication, periodic status checks of neighboring components, and reliable message delivery even when under hostile attack or periods of network congestion or flooding. MCCD Applications use a publish/subscribe paradigm to exchange messages over the IDIP backplane. The IDIP messaging services are shown in Figure 8.

IDIP applications provide intrusion trace, response, and report services within a single community. The ***auditor*** process records connection information at each node. This information is later used by ***idip_processor*** to track an attack back to its source. The ***idip_processor*** also integrates component-specific response functions on each node and manages the local policy files received from the DC.

The DC console provides a display of current network trace activities within a local community, and summary trace information from neighboring communities. Administrators can use DC services to negotiate and implement mutually acceptable cooperation agreements with neighbors.

Automated trace, response, and report actions can be allowed, denied, or approved on a case-by-case basis.

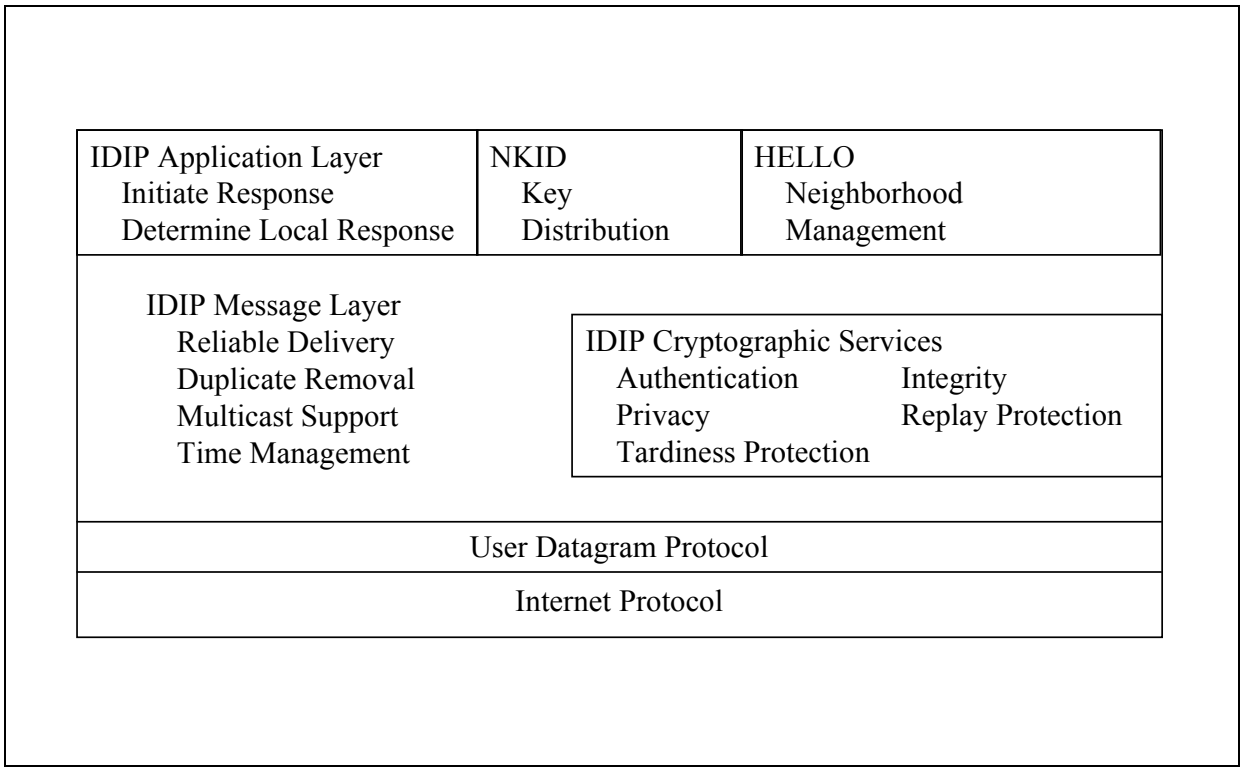

Figure 8 IDIP Backplane Architecture

Tkined, an interactive editor for creating and maintaining network maps, is used to display the local network topology, neighboring network clouds, and current attack-related information. The detecting component, attack source, attack target, and each IDIP component along the attack path are highlighted. ***Tkined*** communicates with other components via the IDIP backplane through the ***dc_if*** helper application. The ***dc_merger*** process reads and combines reports from the backplane before they are sent to the management display.

The ***dc_request*** process listens for policy file request messages from MCCD components and sends the requested file. Each community must have a generic policy file; additional policies may be created for each MCCD component. If a specific policy for the requesting component does not exist, ***dc_request*** will send the generic policy.

2.3 MCCD Management Services

MCCD provides management services to negotiate mutually agreeable sharing and cooperation policies with neighbors and instantiate those policies in local edge boundary controllers. Policies

created at the DC define the information that can be shared with neighbors, any modifications deemed necessary to information received from neighbors, what type of actions to request (trace, respond, report), and what type of actions to provide. These policies are sent to each MCCD component when they are initialized. A new DC_TO_DC_MSG data structure and two new message classes were defined for MCCD management messages.

An administrator can change a policy by using the ***EBCPolicyFileXfer*** process to push a new policy out to an EBC. The ***DCCoopNegotiation*** process can be used by an administrator to negotiate a mutually agreeable policy with a neighbor; once an agreement is reached, both sides instantiate the new policy at their respective EBCs. Negotiation involves offering and requesting any of the following services: (1) intrusion tracing and reporting, (2) intrusion tracing and blocking, (3) propagation to other communities, (4) sharing intrusion alerts, and (5) sharing correlator reports of anomalous events.

MCCD management services use the IDIP DC Application Programming Interface (API) services to receive intrusion reports. The trust management and anomaly correlation software also need this information, so the concept of a DC multicast group was created allowing applications running on different computers to receive the same MCCD management information. This required an addition to the ***network.ini*** configuration file to list the DC nodes and to associate them with a new DC group.

Edge Boundary Controllers (EBCs) are also identified in the ***network.ini*** configuration file, stored on the DC. Each EBC is defined by its internal and external network address. Remote neighborhood groups are created by associating a group network address with the set of external network addresses that belong to the group. A remote neighborhood group will typically list the address of one local EBC, and one or more remote EBCs representing communities that can be reached through the local EBC.

2.4 Cross-Domain Information Sharing

Intrusion trace messages may contain information deemed sensitive to the originating organization. While it may be acceptable to share this information within an organization, policies may limit the sharing of detailed information about network topology (host addresses along the attack path), detection capabilities, or perceived severity of an attack. MCCD services at the DC and EBC enable organizations to manage IDR information content shared with, and services performed on behalf of other organizations. Fields in an incoming or outgoing trace message can be purged, set to an administratively assigned value, or passed unmodified.

A policy language specification was defined to express intercommunity relationships. These policies are stored at the DC in a new ***ebc_msg_policy*** file and sent to each EBC on initialization. This file is organized into sections defining community, outbound trace, inbound trace, EBC actions, and message translation. This is described in [1].

The ***idip_ebc_relay*** process runs on each EBC and exchanges MCCD messages with neighboring EBCs. Policies received from the local DC are enforced by each EBC, ensuring local control of information and services. Actions that require administrative approval are temporarily suspended while a request for authorization is sent to the local DC. The ***DCTraceEscalationProcessor*** presents a window to the administrator (at the DC) describing the requested action and displaying the current policy. The administrator can accept or reject the action, and also has the opportunity to change the policy before sending the response back to the local EBC.

2.5 Multi-Domain Trust Model

Information received from external communities is used to track down attacks and modify the security posture. Mission effectiveness could be compromised if false information is received due to compromises to external nodes. MCCD includes a service that monitors the level of attacker activity in neighboring communities and recommends changes to the services and information shared with those communities when attack thresholds are crossed.

A trust event specification language [1] was developed and used to define a trust policy. To improve performance, trust policies are compiled and linked with custom input-output routines and with the trust event correlation engine. The trust event correlator receives reports of attacks from the IDIP backplane and associates those attacks with neighboring communities where the attacks occurred (i.e., the attack source, target, or path was reported to be in a neighbor). Cooperation levels were reduced when the attack activity in a neighboring community exceeded a predefined threshold. A cooling function was included in the trust policy so that periods of reduced attacker activity could be identified and previous cooperation levels could be restored. This trust policy demonstrated the basic multi-domain trust model capability; a more sophisticated policy that analyzed more complex trust indicators could be developed and linked into the trust event correlator to meet the needs of an operational community.

2.6 Real-Time Strategic Correlation

Previous work on port scan detection consisted of network sensors watching for a minimum number of probes over a fixed time window. Scanners could go undetected by spacing individual probes so that the frequency was below the sensitivity of current detectors. While the detectors could be tuned to detect slower scans, this also increased the frequency of false reports. MCCD developed a sensor to monitor the relative probability of different types of network traffic, and to report low-probability events. These reports are correlated to detect stealthy network scanning activity.

The Statistical Packet Anomaly Detection Engine (Spade) is an anomaly sensor plug-in for the Snort IDS developed for use with Spice correlation engine. Spade computes an anomaly score for each packet based on the observed history of the network traffic. The fewer times that a particular kind of packet has occurred in the past, the higher its anomaly score will be. A

probability table (Figure 9) is maintained that reflects the occurrences of different kinds of packets over time, with a higher weight assigned to more recent events. Entropy estimates are used as a means of gauging the anomalous-ness of events. At any given time, a reporting threshold is defined for the sensor. For each event that exceeds this threshold, an alert is sent to Spice for further analysis.

The full joint probability distribution of events (based on packet information such as IP address, ports, flags, and time) is not feasible to measure, so we investigated using static Bayes network models of the packet probability distribution to decide what was anomalous. After analyzing several different criteria, we found that using ***P(dest IP, dest port)*** provides good results for detecting port scans. Different criteria may be useful for identifying other classes of stealthy attack activities.

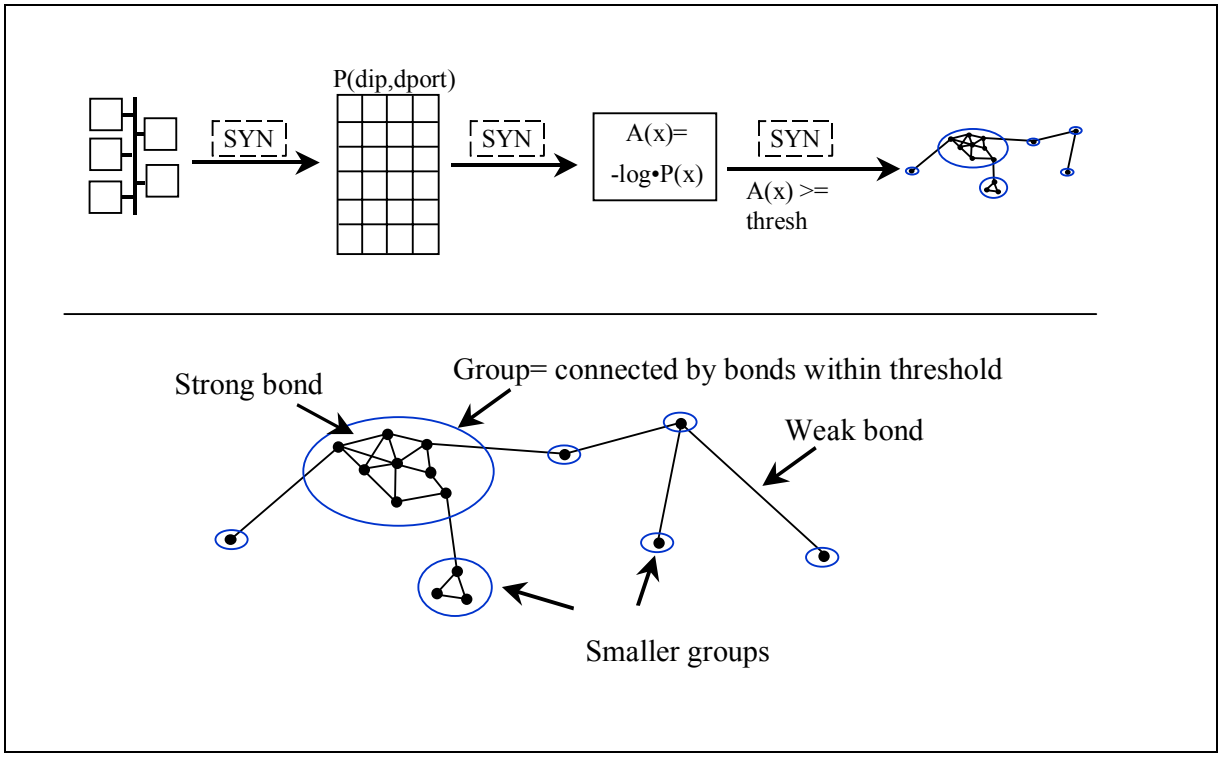

Figure 9 Spade Anomaly Sensor

The Stealthy Portscan Intrusion Correlation Engine (Spice) supports real-time strategic correlation by detecting low-level, widespread attacker reconnaissance gathering activities. Spice detects low level scanning activities by correlating anomaly reports from Spade anomaly sensors within its local community, or from other correlation engines in neighboring communities. This hierarchical analysis approach minimizes bandwidth requirements by filtering both normal activities and anomalous events that have been classified by the correlator;

only reports of anomalous activity that has not yet been classified are forwarded for further analysis.

Spice is composed of multiple computing threads that receive and evaluate anomalous event reports, and generate intermediate correlation reports and correlated scan reports that could be used by administrators or other MCCD components (Figure 10). Spice builds and maintains a correlation graph by calculating the strength of bonds between nodes using "equal" and "close" heuristic evaluation functions; custom heuristic functions can be easily added. New events are added to the correlation graph by choosing random initial bonds and improving these with simulated annealing. Correlation groups are identified by finding all events that are connected in the graph by a bond that exceeds a selected strength. Scan alerts or correlation reports are generated when a group is found with a high combined anomaly score. Graph maintenance threads handle deleting old event, forming bonds to reconnect the graph if needed, and discarding weak bonds not needed to maintain overall connected-ness.

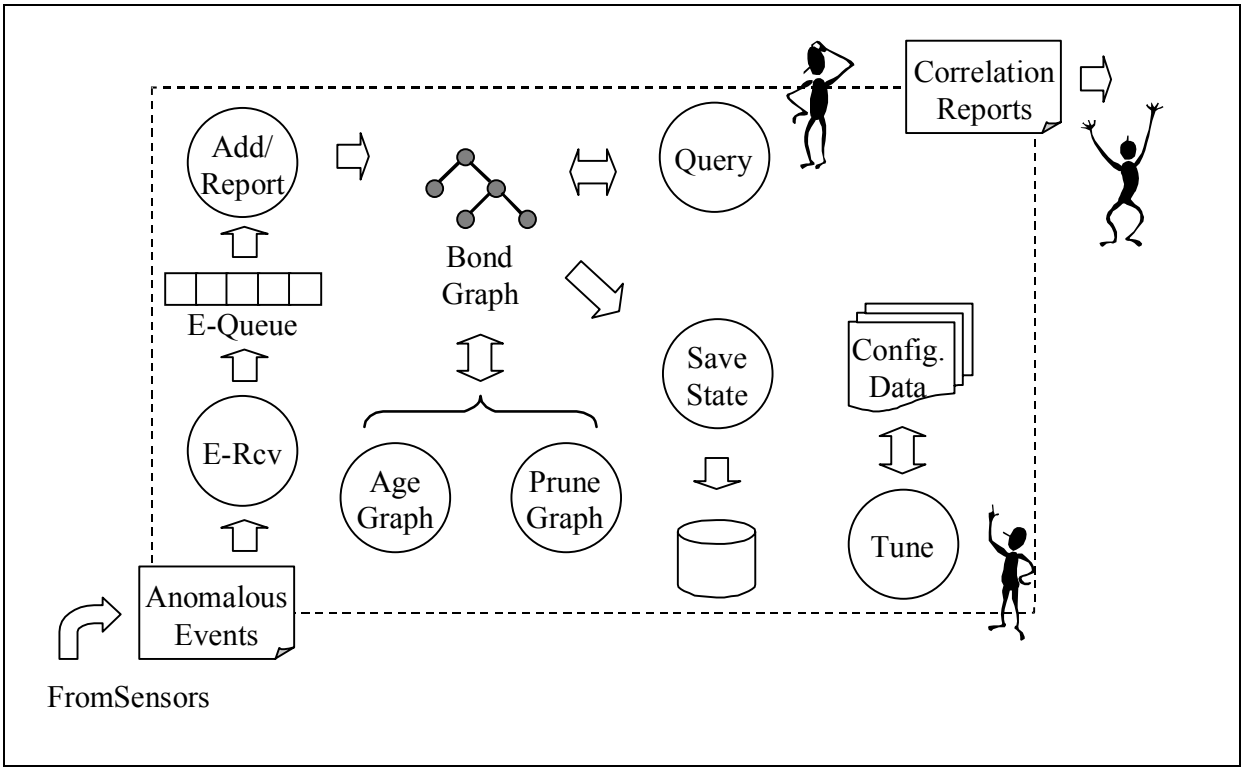

Figure 10 Spice Correlation Engine

Reports of intrusions generated by a correlation engine presented a new challenge to the tracing service provided by IDIP. Spice correlates information received from one or more Spade anomaly sensors that may be located in other IDIP neighborhoods; therefore the correlation engine (and its IDIP neighbors) might not be on the attack path and cannot begin tracing the

attack locally. When a correlated attack is reported, the ***dc_trace_from_here*** process enables an administrator to craft an IDIP trace message and send it to an IDIP component known to be on the attack path. The IDIP component will then begin the trace action.

Reports of active scans are sent to the local DC where they may be shared with other organizations. The ***mccd_alert_sharing*** function enables Spice to send alerts (information about anomalous events that have not yet been correlated to an active scan) to a neighboring community where they may be processed by another Spice correlation engine. This approach was shown to be effective at detecting very stealthy scanning activity distributed over multiple communities. Without the sharing and aggregation of this information, these scans would have gone undetected.

3.0 PROGRAM ACCOMPLISHMENTS

The following sections detail this project's major accomplishments, specific capabilities developed, and lessons-learned.

3.1 Overall Accomplishments

The following were the top-level program technical objectives:

a. Intrusion correlation techniques and tools that scale up to regional and national levels.

   Detection of stealthy, widespread reconnaissance activities requires collection and analysis of a tremendous amount of data gathered from a large number of computers. Effective identification of this class of activity must be done at a regional or national level where information can be correlated from numerous scanned systems. MCCD correlation techniques minimize network bandwidth and processing requirements of hierarchical analysis systems by data filtering and analysis techniques designed to eliminate both common and correlated events from the analysis stream.

   Spade anomaly sensors maintain information about "unlikely" events while discarding common events, effectively reducing the storage and reporting requirements close to the detection point. Spice correlation techniques applied at the local community level effectively combine anomaly reports and generate intrusion (scan) reports. Local Spice correlation engines can export state information about unresolved anomalous events to regional analysis centers where regional Spice correlation engines can analyze information from several communities. Information about relatively common activities and identified scanning activities are not forwarded, reducing the network, storage, and processing requirements at each level. Unresolved correlated events at the regional level can be exported to the national analysis centers for further analysis by a Spice correlator.

b. A trust model for intrusion detection and response (IDR) across disjoint administrative domains, with techniques for assessing trust.

   A simple trust model was developed to demonstrate the trust event correlator (TEC). This model assumes that the degree of cooperation and trust appropriate for each neighboring community is inversely proportional to the frequency of attack activity occurring in each neighboring community. The TEC receives copies of attack reports sent to the DC and associates each attack with neighboring communities that were on the attack path (source, destination, or observer). A trust policy change recommendation is generated when the current attack threshold set for a given neighbor is exceeded. Periodic events are injected into the event stream causing the model to reduce the recorded attack levels by a defined percentage. This allows the model's current attack levels to decrease during periods of

inactivity; a trust policy change recommendation is generated when the current attack threshold drops below predefined levels.

c. Capabilities required for survivable, cooperating IDR systems across organizational boundaries.

   Cooperation between organizations is necessary to effectively detect and defend against multi-staged network attacks since each domain has incomplete information. The attacking computer may reside outside of the attacked domain, leaving the victim helpless to stop the attack; the attacking computer may reside in a domain that is not aware of its hostile activities. Implementing the IDIP trace and response services within each organization enables them to effectively trace and block attacks that they detect; MCCD enables them to notify neighbors that they may be involved in an attack, enabling the neighbors to investigate the attacker activities and take appropriate action.

   MCCD information is exchanged between edge boundary controllers (EBCs) in neighboring community. Pair-wise relationships between EBCs are administratively defined; cryptographic session keys are exchanged, and periodic "hello" messages are exchanged to insure liveness. EBCs receive local IDIP trace messages describing attacks that they might have seen. Trace messages are forwarded to EBCs in neighboring communities if (1) the local EBC observed the attack, and (2) the local policy authorizes sending trace messages to the neighboring community. Before forwarding the trace message, an EBC may sanitize some fields to hide sensitive information that could reveal (1) internal network topology, (2) local detection capabilities, or (3) local severity assessments. Local administrators use policies to define field sanitization and services requested for outgoing messages, and field translation and services honored for incoming messages.

3.2 Capabilities Developed

The following list summarizes the work completed under this program:

a. Developed the initial MCCD requirements and its operational concept, documented in [2]. The concept of operations provides a hypothetical scenario to both illustrate the concepts of distributed, coordinated intrusion response, and to provide a framework for evaluating the effectiveness of the MCCD concepts. This challenge problem describes how the MCCD elements could react to a distributed denial of service attack proceeded by low-frequency resource mapping and exploits spanning multiple domains and service providers.

b. Identified IDIP enhancements and new services needed to meet the MCCD requirements, documented in [1]. This document includes a detailed breakdown of the components necessary to develop an MCCD system, describing the system's interfaces by defining the messages that are exchanged between communities, messages that are exchanged between components, within a single community, and administrator interfaces and configuration files.

This document also discusses operational constraints when operating with multiple classification domains or coalition forces.

c. Modified the IDIP software to support MCCD requirements. The IDIP messaging layer (***idipd***) provides a secure, reliable, and survivable publish/subscribe service that implements the IDIP protocol. IDIP applications monitor network traffic (***auditor***), provide integration with existing sensors and detectors (***idip_reader***), and perform event-tracing and component-specific response functions (***idip_processor***). IDIP management processes provide a security management graphical interface (***tkined***), combine related intrusion reports into a single report (***dc_merger***), serve configuration and policy files to MCCD components (***dc_request***), and export copies of reports in a variety of formats (***dc_reports***).

d. Developed new MCCD management software components. The IDIP management backplane was enhanced to support a multicast service allowing the management applications to run on different computers. The ***network.ini*** file is now used to define the edge routers (EBCs) within a community, and to define groups of EBCs (local-remote pairs) that represent community relationships. A new ***ebc_msg_policy*** configuration file was defined to describe IDR information sharing and cooperation policies. Both generic and component-specific policy files are supported. ***EBCPolicyFileXfer*** is used to send policy files to EBCs where they are received by ***idip_processor***. A new negotiation service (***DCCoopNegotiation***) was developed to aid administrators trying to establish a mutually acceptable policy between two communities. Alerts and reports can be shared with neighboring communities or regional analysis centers using the ***mccd_alert_sharing*** service.

e. Developed new MCCD communications software components. New identifiers were added to the IDIP and CIDF messages to mark messages entering a community, and a new IDIP_DEVICE_EDGE type was created to identify EBCs. ***Idip_processer*** was modified to apply policy rules to Trace messages received at an EBC, including message translation, message sanitization, and trace escalation (a new feature where a trace is temporarily suspended while authorization is requested from an administrator).

f. Developed inter-community cryptographic services. The IDIP Network Key Information Distribution (NKID) protocol was adapted to provide session key exchange between communities. The Community Key Information Distribution (CKID) protocol uses an asymmetric cryptographic system to verify the authenticity of neighboring communications and to exchange new cryptographic keys to be used for future message integrity and privacy.

g. Developed trust model and trust event correlator. The Trust Event Correlator (TEC) uses a compiled trust policy (event handler), custom I/O software, and a generic correlation engine to evaluate the "trustworthiness" of remote communities. A generic trust policy language and compiler were developed to create specific trust policies. A simple policy was created to measure the frequency of attack activity in each neighboring community; recommendations to adjust IDR information sharing and cooperation levels are issued when thresholds in the model are reached.

h. Developed stealthy event detection sensor (***Spade***) and correlation engine (***Spice***). The Spade anomaly sensor observes network traffic and calculates an anomaly score that reflects the occurrences of different kinds of packets in history, with a higher weight assigned to more recent events. A report is generated if an event exceeds a preset anomaly threshold. The Spice correlation engine adds Spade alerts to a bond graph linking together events related by a set of heuristics. The Spice implementation uses heuristics tuned to detect slow, distributed port-scanning activities. Weak links are pruned from the graph, while strong clusters are reported as detected port scans. A Spice engine running in a regional or national analysis center can import Spice bond graphs from multiple communities (using the ***mccd_alert_sharing*** process) for early detection of widespread scanning activity that would go undetected in each local community. A new administrative function was required to initiate a trace action when the detector is not on the attack path. This situation may occur when the Spice correlator identifies a scan based on information received from Spade sensors. A trace can be initiated at one of the reporting sensors that observed the attack traffic by an administrator using the new ***dc_trace_from_here*** process.

i. Integrated components into a multi-community lab for concept validation testing. An initial proof-of-concept demonstration was provided to show the basic trace, response, and reporting features working between two communities. Different static policies were used to show both automated cooperation between communities, and administrator authorization of inter-community actions. The Spice correlation engine was shown to be able to detect stealthy port scanning activities. A more complex configuration of independent network communities was used in the final demonstration to illustrate tracing, responding, and reporting through multiple communities. The cooperation negotiation feature was used to negotiate policies between two administrators managing separate communities. The trust event correlator was used to monitor attack frequency and issued policy change recommendations. Some initial integration with a course of action generator (Propheteer) was demonstrated, showing that information can be exported and imported from MCCD using the IDMEF message standard [6]. Shared correlation reports validated that Spice could use information from other communities (i.e., other Spice correlators) to significantly reduce the time required to detect stealthy scanning activities. In a separate activity, the Spice correlator was evaluated against the Snort portscan detection plug-in and shown to significantly outperform the Snort portscan detector at detecting port scans. Spice detected probes spaced up to four hours apart, while the Snort portscan plug-in had difficulty accurately detecting probes spaced more than one minute apart.

j. Developed instructions for configuring and running the MCCD software, documented in [3]. The Users Guide includes a brief overview the MCCD architecture, descriptions of the major functional subsystems, including “man”-style manual pages for each executable process, and provides details on how to run each software component. A worked example is included consisting of two separate network domains that share information and cooperate in tracking down and responding to intrusions. This report also includes an analysis of the bandwidth

requirements of MCCD-initiated network traffic and recommendations for further minimizing the impact of IDR traffic on network performance.

3.3 Lessons Learned

The following paragraphs summarize key lessons-learned from this effort.

a. **IDIP integration and performance impact**. The core IDIP software proved easy to extend, providing the basic trace, response, and reporting services required for MCCD. New features were easily added by creating new processes, defining new IDIP message types, and using the IDIP publish and subscribe services to exchange messages. The network overhead of the IDIP protocol was examined in terms of fixed bandwidth requirements due to initialization communications and liveness tests, and variable bandwidth requirements due to attack tracing and reporting activities. Replacing the variable-length "hello" message with a fixed size message could reduce the fixed overhead, since the variable-length fields are never used. Neighbors exchange "hello" messages every 300 seconds; the protocol overhead could be reduced by 50% if "hello" messages were sent only when nodes haven't heard from a neighbor in the past 300 seconds. The variable bandwidth requirements are determined by the number of attacks that are detected, the number of detected attacks that are traced (determined by local policy), and the number of nodes that are involved in each trace (e.g., nodes on the attack path). Bandwidth usage can be managed by minimizing the number of nodes in each IDIP neighborhood, by judicious selection of traceable intrusion events, and by blocking or detecting attacks close to community boundaries whenever possible. This analysis is documented in [3].

b. **IDMEF message integration**. One goal of this project was to simplify integration with other products by using standards-based protocols and message formats. IDIP used the CIDF standard to achieve this goal; however, IDMEF is the current standard supported by the IETF for exchanging reports between IDS components [6], [7]. While IDMEF is able to describe an intrusion event, it currently does address the intrusion tracing, response, and reporting requirements needed for MCCD. For this reason, the IDIP/CIDF message structures were retained for communicating between MCCD components. Routines were developed to import and export messages in the IDMEF format; however liberties were taken with the IDMEF ADDITIONALDATA class to include required information. This approach reduced the interoperability advantages of using a standard. The Snort correlator produced and consumed reports using the IDMEF format, which simplified correlation report sharing between communities. Integration between MCCD and the Propheteer course of action generation software was also accomplished using IDMEF; however the type of information expected by Propheteer was at a significantly higher level of abstraction than the IDS concepts that can be expressed by IDMEF, which limited the usefulness of this integration effort.

c. **Integration with higher analysis functions**. Although this was not an initial requirement for MCCD, it was desirable to develop methods for integrating MCCD with other types of

network management and security analysis tools. Trying to identify other DARPA Cyber Panel projects for integration was difficult do to incompatible prototype development schedules, the varying degrees of maturity of each project, and disparities in the level of abstraction each project addressed. Some degree of integration was achieved between MCCD and the Propheteer course of action generation project, enabling MCCD to report a specific kind of network attack, and Propheteer to present an operator with a limited set of options for handling the event. The primary difficulties encountered were conceptual; MCCD deals with an attack event, an attack path, and the responses taken to block the attack, while Propheteer deals with impact to mission due to compromises of critical resources. Effective integration would have required an additional component that understood the mission of an enclave, the critical resources within an enclave, and the effect that MCCD-reported attacks against those resources have on performing the mission. A process able to map Propheteer-suggested courses of action to MCCD-implementable responses would also be desirable in an integrated system.

d. **Community auto-discovery**. The current implementation requires that MCCD communities be configured with the addresses of all neighboring communities. In addition, public keys must be pre-exchanged to facilitate the initial CKID exchange. This approach works in static environments where all neighboring communities are known, but breaks down when a community is connected to an Internet cloud. Either the community must establish relationships with all other communities connected to their ISP, or the ISP must become an MCCD community. The addition (or deletion) of a community requires all neighbors to update their configuration information. A better approach would be to establish a directory service or trusted authority that would register new communities when they come online, and provide this information to communities tracing an attack originating beyond their borders.

e. **Delegation of trust**. The MCCD architecture enables IDR information sharing and cooperation between mutually competitive organizations by retaining independent management of internal information, resources, and delegation of trust. Trust can be tightly held within an organization by requiring administrative approval of all MCCD actions, or can be selectively delegated to EBCs where local policies are enforced. Delegation of trust is further extended by conditionally providing information to neighboring communities; it may be desirable to provide information but request that it not be shared with other organizations. The transitive relationships created by the introduction of intermediate communities, and verification of the implied trust requires further exploration.

f. **Detection off the attack path**. The concept of IDIP neighborhoods enable a small number of IDIP-enabled components within a community to actively trace an attack by starting from the node that detects the attack, and identifying those nodes in its neighborhood who observed the same network traffic. Each observer repeats this process until the source of the attack is located. This process fails if the detecting component is not on the attack path; in this case, none of its neighbors observed the network traffic. This can occur when detection is made by a component analyzing information from nodes that are on the attack path. Spice

detects attacks by analyzing information received from Spade sensors, but may not directly observe the attack activities. A new service was needed to craft a trace message and send it to one or more components believed to be on the attack path asking them to initiate the trace action.

g. **Community resistance**. ISPs are fearful of lawsuits and will resist *official* exchanges of information among their organizations. They will only cooperate with law enforcement if presented with a court order. However, information is exchanged *informally* among individual administrators when networks are down or other problems occur. Widespread cooperation is most likely to occur within organizations, such as among branch offices that belong to a single corporation. Widespread acceptance of automated tracing and information sharing will require changes to current business environments that cannot be achieve through technology, alone.

4.0 FURTHER INVESTIGATIONS, RESEARCH, AND DEVELOPMENT

The following additional work is recommended to improve MCCD functionality in operational environments:

a. Standards development. Interoperability demands standards, and while IDMEF achieves its goal of exchanging intrusion reports, it does not yet address intrusion tracking or intrusion response. These concepts, which have been explored with IDIP and MCCD, should be added to IDMEF [6], [7], or a related standard [10], to gain acceptance from commercial vendors.

b. Administrative work policy analysis. Understanding how network operation centers identify and respond to intrusions may identify changes to MCCD functions and user interfaces that could improve administrator workflow.

c. Integration with network and security management products. Network management products used to monitor equipment and maintain system services could be integrated with MCCD to help administrators determine if detected problems are hardware-related or caused by a malicious user. Integration with emerging security products could provide additional situational assessment, mission impact, and recommend course-of-action alternatives to maintain critical services.

d. Implementation of distributed management services with fail-over mechanisms. The MCCD distributed backplane enables management services to run on different computers. Applying fault tolerant techniques to create warm spares of critical management services, with fail-over and recovery mechanisms could further enhance survivability by eliminating single-point failures. The concept of multiple redundant EBCs between adjacent communities should also be explored.

e. Software hardening. The current software proved valuable for prototyping the MCCD concepts and validating their usefulness; however, additional work is needed to harden and test the code so that a fielded IDR system can withstand directed attacks against MCCD components.

f. Classification domain boundaries. The concept of providing MCCD services between coalition networks or classification domains was explored, and key issues with assurance and downgrading were identified. Integration with a commercial Guard is needed to validate that the MCCD protocol will work when a trusted Guard (e.g., high-assurance MLS firewall) with accredited downgrade rules is placed between two neighboring EBCs.

g. Auto-discovery of neighboring communities. A protocol is needed to enable an EBC to locate neighboring MCCD-enabled communities that may be helpful in tracking down an intrusion. A default paranoid policy that disables sharing and does not forward or honor trace requests could be instantiated for newly discovered communities until administrators in both communities negotiate a mutually acceptable policy.

h. Detecting additional stealthy attack activities. Spice may be able to detect additional stealthy activities by using different heuristics to correlate a different set of packet characteristics. Distributed Denial of Service (DDoS) control networks represent one class of stealthy activities that merit further analysis.

i. Analysis of different trust policy models. Further work is needed to identify techniques for evaluating the "trustworthiness" of neighboring communities. Once identified, the MCCD trust model could be extended to better evaluate the current state of neighboring communities and make better-informed recommendations for adjusting MCCD policies.

j. Analysis and potential integration of emerging security protocols for multicast services [8], [9] to supplement or replace the current NKID and CKID implementations.

5.0 EXPERIMENATION AND DEMONSTRATION RESULTS

5.1 Capability Demonstrations

Figure 11 shows the configuration used in the final proof-of-concept demonstration. The demonstration objectives were to (1) validate that MCCD components can trace intrusions over community boundaries and implement blocking actions, (2) that administrators can negotiate and manage the information shared with, and services provided to neighboring communities, (3) that policies can be dynamically changed by an administrator, (4) that the trustworthiness of neighboring communities can be monitored, (5) that stealthy port scanning activities can be detected, and (6) that reports of intrusion activities can be shared among communities. This environment provided a reasonably complex network for testing MCCD, and introduced the transitive relationship of tracing and reporting through an intermediate community. During the testing, several implementation errors were found, but no flaws in the basic concepts.

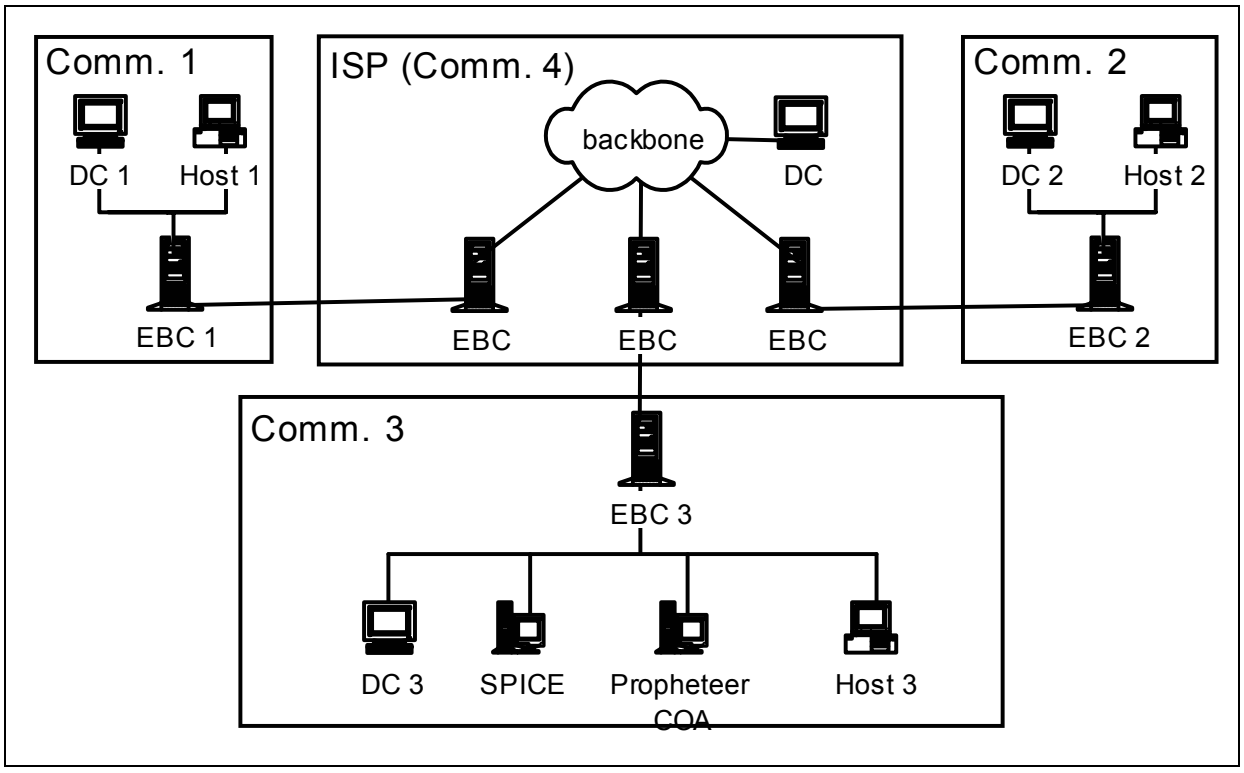

Figure 11 Demonstration Configuration

An interim capabilities demonstration was conducted using three independent communities interconnected by a simulated wide area network to test initial tracing and policy capabilities. One community pair implemented a cooperative policy with automated tracing and reporting, while a restrictive policy was implemented between another community pair where

administrative approval was required prior to any cooperative tracing. A commercial IDS and a Spade/Spice detector were included in one of the communities to compare detection of stealthy scanning activities initiated from the other two communities.

The following capabilities were shown in the interim capabilities demonstration

a. Attacks between two communities were traced, blocked, and reported.
b. Interactions (trace, response, report) were controlled by local policy.
c. Message filtering and translation was implemented at the community boundary
d. Trace authorizations were escalated to an administrator for approval.
e. Dynamic policy modification and update were performed by an administrator
f. Stealthy port scans were detected by Spade/Spice, but missed by the COTS detector

The final capabilities demonstration architecture replaced the simulated wide area network with a fourth community to demonstrate tracing and reporting through intermediate communities. A Spade/Spice component was added to a second community to show improved scan detection when intermediate results from one community are combined with information from another community, the trust event correlator was introduced and demonstrated, and a Propheteer COA component was added.

The following new capabilities were shown in the final capabilities demonstration

a. An attack involving three cooperating communities was automatically traced and blocked.
b. An attack involving two non-cooperating communities was successfully traced and blocked after administrators negotiated the trace and response services. These services were later renegotiated to require authorization for some actions.
c. All communities involved in an attack received reports and were able to display the attack path. Each community displayed their view of the attack, including neighboring community network clouds and local hosts and routers that were involved in the attack as detectors, attacker, victim, or observers. An unexpected result was that a community not involved in the attack was able display that its neighbor was under attack after receiving a trace message.
d. Improved detection with correlation sharing. One community detected a moderate, distributed scan in 54 minutes. The same scan was detected in 23 minutes when information from a second community was shared. The IDMEF message format was used for exchanging correlation reports.
e. Trust event correlator recommended raising the “INFOCON” level when attack frequency increased, and recommended lowering the “INFOCON” level after a period of inactivity.

f. Propheteer provided course of action recommendations when an unauthorized root escalation was detected on a mission-critical computer. The IDMEF message format was used for reporting the root escalation attack to Propheteer.

The demonstrations successfully showed that using MCCD to defend against attacks over a complex network of networks is feasible, even when communities employ different policies. Shared information provided early detection of stealthy reconnaissance activities and a broad understanding of remote and distributed attacks. Additionally, automated cooperation facilitated rapid response to intrusions.

5.2 Stealthy Portscan Experiment

The validation demonstrations showed that the Spice correlator could successfully detect stealthy port scans. An experiment was run to (1) compare optimal detection capabilities of Spice with another popular port scan detector, and (2) to measure the sensitivity boundaries of Spice. The Snort portscan detector was selected for comparison.

Objective - The goal of the experiment was to compare the detection performance of Spice and Snort's standard portscan detector in different situations, and to measure the sensitivity boundaries of Spice. Several test runs were conducted using different scan configurations. The configuration of each detector was adjusted between scans in an attempt to maximize their detection performance. An ideal detector would report all probe packets together as a scan with out including any extra, non-scan packets.

Although the detectors monitored equivalent scans, direct comparison between two detectors under identical circumstances proved difficult, as the detectors each had different operational parameters that could be tuned to improve detection. Both the scan configuration and the IDS configuration had to be considered.

Metrics - The following metrics were originally defined for this experiment; however due to time constraints, only efficiency and effectiveness were calculated from the experimental data.

- Efficiency – fraction of scan events reported
- Effectiveness – fraction of events reported that are scan probes
- Report Effectiveness – fraction of reports that contain at least one scan probe
- True report cohesion – how close are the scan probes to being all in 1 report?
- True report non-noise - # reported scan probes / # of events in true reports

Configuration - The scan configurations consisted of a single SYN packet to port 109 on each of 100 destination IP addresses randomly chosen from a /24 network block. The source addresses for the scan packets were chosen from a random block of 1, 5, 20, or 100 IP addresses. Delays between packets from any source were configured for 1 second, 5 seconds, 1 minute, 5 minutes, 1 hour, or 4 hours. Each scan was centered within 3 weeks of traffic collected from a small business; the traffic addresses were remapped to the new test network. Five different

random seeds were used so that multiple samples could be generated from otherwise identical configurations.

Results - Experiment results show that the Snort portscan detector is able to detect scans with small inter-packet gaps (e.g., less that 1 minute between scan packets). The configurations tested did not detect scans when the delay between scan packets was greater than one minute. Under the same scanning conditions, the Spade/Spice configuration was able to detect all scans with inter-packet gaps of one minute or less. Subsequent testing showed that Spade/Spice was also able to detect scans with gaps of 1 hour, but the efficiency and effectiveness decreased when the gap was increased to 4 hours. Further analysis of the data is needed to compare the report effectiveness, true report cohesion, and true report non-noise characteristics for the two detectors.

Further testing and analysis of the Spice correlator should be conducted to determine if there is a fundamental characteristic that causes detection to break down around the 4 hour inter-packet gap. Understanding this characteristic could lead to the development of more sensitive port scan detectors.

5.3 Operator Validation Survey

The MCCD cross-domain traceback and response concepts were developed to improve incident handling by automating coordination between organizations currently done by network and security administrators. These are the people who would use MCCD in an operational setting, so their acceptance is critical to eventual deployment of this, or similar technology. Three people having backgrounds as a Lab Network Manager, a manager of a Security Operations Group of a large ISP, and a university Network Administrator, were provided with background information about MCCD concepts and a demonstration of MCCD capabilities.

The network managers/administrators gave real world, practical feedback on the technology developed for the MCCD program. They felt the ideas explored in this project are important and should continue to be developed. To make this technology better, they suggested adapting it to use new protocols that are beginning to emerge from within the IETF and getting involved in the IETF to influence the evolution of those protocols. Another suggestion was to improve the security of the management station, since if it is compromised it could provide a path to another community. An auto-responder for cooperation negotiation is an important feature that should be prototyped, since operators may be overwhelmed with too many cooperation negotiation and escalation requests. Default settings, including for an auto-responder, should be to sanitize as much information as possible. Other suggested enhancements included sending an acknowledgement when a request has been presented to the other community and having the option to suppress reporting back the results of a traceback request. Since cooperation among ISPs is not likely to be widespread, additional thought should be given to using this technology between pairs of end-point organizations that have business relationships or among networks that belong to a single organization such as branch offices that belong to a global corporation. These suggestions may allow MCCD technology to become more practical and acceptable to network

operators. Nevertheless, the largest remaining obstacle to the deployment of such technology is the cultural resistance, especially among ISPs, to open cooperation in identifying and shutting off the sources of network attacks.

6.0 SUMMARY AND CONCLUSION

This program demonstrated the feasibility of using a local policy-based approach to manage intrusion tracking and response services that are coordinated between multiple organizations. This work allows independent organizations to track down intruders that traverse network boundaries and block their activity, while protecting locally-sensitive intrusion-related information and managing intrusion tracing and response services.

A new anomaly correlation approach based on bond graphs that grouped anomalous activity was developed and demonstrated to be effective at detecting stealthy port scanning activities that went undetected by existing techniques. When integrated with the MCCD report sharing service, it was shown that when the bond graphs from multiple organizations were collected and correlated, widespread, sparse stealthy scanning activities that went undetected in a single organization could be identified.

6.1 Recommended Future Work

There are several areas where MCCD concepts and the current implementation require additional work prior to widespread use. Section 4.0 identifies recommended changes to improve survivability in hostile environments through software hardening and fault tolerant fail-over mechanisms. Current standards work in the areas of Intrusion Detection [6], [7], [10] and Multicast Security [8], [9] may improve interoperability and security of MCCD messages. Integration with existing network management components and better user interfaces designed to improve operator workflow are needed. New techniques to identify mission dependencies on computing resources, determine mission impact when resources are under attack, and provide course-of-action recommendations to detected (or predicted) attacks are needed to understand and defend against strategic attacks. Finally, additional research in the areas of strategic level correlation and in remote community trust determination is needed.

6.2 Conclusions

We have developed and demonstrated a concept for cooperative intrusion detection and response across small- to very-large-scale networks of networks spanning numerous administrative domains. A flexible *cooperation* policy definition language and enforcement mechanisms were developed, enabling each organization to maintain local control of incident information, tracking services, and intrusion responses. Today's complex formal and informal organizational relationships can be expressed through hierarchical and peer-to-peer policies that describe varied cooperation and reporting relationships. A novel technique for analyzing anomalous network traffic was developed and shown to be effective at identifying slow, stealthy network scanning activities. Techniques for sharing intermediate correlation results among organizations proved effective at detecting widely distributed stealthy scanning activities that were undetected at the local level.

The mechanisms developed support the original MCCD requirements, as well as providing better control over intrusion response.

a. **Operating while the system is under attack.** The use of a lightweight, secure, reliable UDP for communication reduces the effects of attacks on MCCD traffic.

b. **Autonomous response.** The mechanisms defined allow each MCCD community to independently determine their responses based on the MCCD messages and local policy parameters.

c. **Detecting stealthy reconnaissance activity.** The correlation techniques performed better than expected at detecting stealthy port scans. While there is much debate over the threat represented by a port scan, awareness of stealthy reconnaissance activities should be part of any comprehensive network defense strategy. The techniques developed may be applicable for detecting other classes of stealthy attack activities; further research is needed in this area.

d. **Minimal system performance impact.** After initialization, IDIP consumes minimal network bandwidth when there is no attack activity. MCCD introduces little additional overhead; only infrequent keep-alive messages are used to maintain the community state. During attacks, use of relatively compact messages minimizes affects on network resources. Policies enforced at community boundaries drop unwanted IDR traffic.

e. **Measuring community trust.** We found fewer alternative trust indicators that could be used to gauge the current state of remote communities than originally anticipated. Additional techniques and mechanisms are needed to accurately gauge the state of cooperating neighbors. A related question deals with appropriate policy changes in response to detected changes in trust. If a neighboring community is under active attack, should additional assistance be provided to aid in defending the neighbor, or should less information be provided because it may fall into the hands of the attacker?

7.0 REFERENCES

1. The Boeing Company. *Multi-Community Cyber Defense (MCCD) Design Report*, Boeing Document Number D658-10968-1, January 2002.

2. The Boeing Company. *Multi-Community Cyber Defense (MCCD) Requirements and Concept of Operations*, Boeing Document Number D658-10964-1, July 2001.

3. The Boeing Company. *Multi-Community Cyber Defense (MCCD) Users Guide*, Boeing Document Number D658-10969-1, October 2001.

4. The Boeing Company. *Protocol Definition Intruder Detection and Isolation Protocol Definition*, Interim Technical Report, Boeing Document Number D658-10732-1, January 1997.

5. The Boeing Company. *Dynamic Cooperating Boundary Controllers*, Final Technical Report, Boeing Document Number D658-10822-1, February 1998.

6. IETF Intrusion Detection Working Group. *Intrusion Detection Message Exchange Format* Data *Model and Extensible Markup Language (XML) Document Type Definition*, Internet Draft, draft-ietf-idwg-idmef-xml-06.txt, D. Curry (Merrill Lynch), H. Debar (France Telecom), 28 December 2001.

7. IETF Intrusion Detection Working Group. *The Intrusion Detection Exchange Protocol (IDXP)*, Internet Draft, draft-ietf-idwg-beep-idxp-03.txt, B. Feinstein (Guardent, Inc.), G. Matthews (CSC/NASA Ames Research Center), J. White (MITRE Corporation), 11 September 2001.

8. IETF Multicast Security Working Group. *Group Secure Association Key Management Protocol*, Internet Draft, draft-ietf-msec-gsakmp-sec-00.txt, H. Harney (SPARTA), A. Colegrove (SPARTA), E. Harder (NSA), U. Meth (SPARTA), R. Fleischer (SPARTA), March 2001.

9. IETF Multicast Security Working Group. *Group Key Management Architecture*, Internet Draft, draft-ietf-msec-gkmarch-01.txt, Mark Baugher (Cisco), Ran Canetti (IBM), Lakshminath Dondeti (Nortel), 23 October 2001.

10. Incident Object Description and Exchange Format Working Group (IODEF WG), http://www.terena.nl/task-forces/tf-csirt/i-taxonomy/index.html

11. Rich Feiertag, Cliff Kahn, Phil Porras, Dan Schnackenberg, Stewart Saniford-Chen, and Brian Tung, "*A Common Intrusion Specification language*," www.gidos.org/, June 1999

Glossary

	Application Programming Interface
PI	
CIDF	Common Intrusion Detection Framework
CKID	Community Key Information Distribution
DARPA	Defense Advanced Research Projects Agency
DC	Discovery Coordinator
EBC	Edge Boundary Controller
IDIP	Intruder Detection and Isolation Protocol
IDR	Intrusion Detection and Response
IDS	Intrusion Detection System
MCCD	Multi Community Cyber Defense
NKID	Neighborhood Key Information Distribution
SPADE	Statistical packet Anomaly Detection Engine
SPICE	Stealthy Portscan Intrusion Correlation Engine
TEC	Trust Event Correlator
TM	Trust Manager